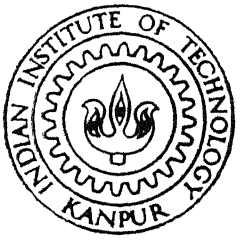
HYPERELLIPTIC CURVES FOR PUBLIC KEY CRYPTOSYSTEMS

by

Nitu Jain



DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

JANUARY 1997

HYPERELLIPTIC CURVES FOR PUBLIC KEY CRYPTOSYSTEMS

A Thesis Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Master of Technology

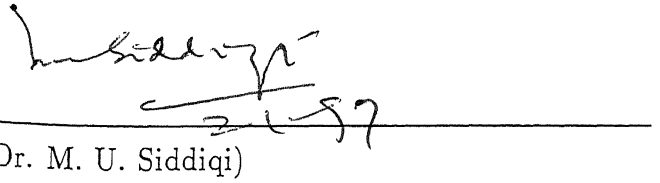
by
Nitu Jain

to the
Department of Electrical Engineering
Indian Institute of Technology, Kanpur
Jan 1997

EE-1997-MTAI-HYP

Certificate

Certified that the work contained in the thesis entitled "*Hyperelliptic Curves for Public Key Cryptosystems*", by Ms. *Nitu Jain*, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.



(Dr. M. U. Siddiqi)
Professor,
Department of Electrical Engineering,
Indian Institute of Technology,
Kanpur.

Jan 1997

Abstract

Security of some of the existing public key cryptosystems has been seriously threatened by the recent advances in computing discrete logarithms and integer factorization. In this thesis, we present the issues related to the implementation of a hyperelliptic cryptosystem. The discrete logarithm problem so far for these systems is intractable. The jacobian of a hyperelliptic curve defined over a finite field forms an abelian group. For a secure cryptosystem, the group order should have a large prime factor. We have selected certain curves and tabulated the order of their jacobian for varying n, when the ground field has characteristic two. An efficient algorithm for computing the multiple of a group element has been presented. Its effectiveness has been shown by tabulating the timing details.The concept of normal basis representation is introduced for enhancing the efficiency. Diffie Hellman key exchange and ElGamal scheme has been described with reference to hyperelliptic curves. Attempt has been made to design a hyperelliptic curve analog of ElGamal scheme, for the case of genus two.

Acknowledgements

I express my sincere and heartfelt gratitude to my thesis supervisor, Dr. M.U. Siddiqi. His esteemed guidance helped me to sail through the toughest of times. Words are not enough to express my thanks for his tolerance and cooperation during the final stages.

I take this oppurtunity to thank my labmates and friends for their invaluable support and cooperation in the work.

I dedicate this work to my family, the heavens of all places, without which I would never have been able to accomplish this task.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Overview of cryptography

The fundamental goal of cryptography [6, 17] has historically been to acheive privacy, i.e., to enable two people, A and B, to send each other messages over an insecure channel in such a way that only the intended recipient can read the message.

A *cipher* is a secret method of writing whereby *plaintext* is transformed into *ciphertext*. The process of transforming plaintext into ciphertext is called **encryption** while the reverse is called **decryption**.

Cryptanalysis is the science and study of methods of breaking ciphers. The branch of knowledge embodying both cryptography and cryptanalysis is called **cryptology**.

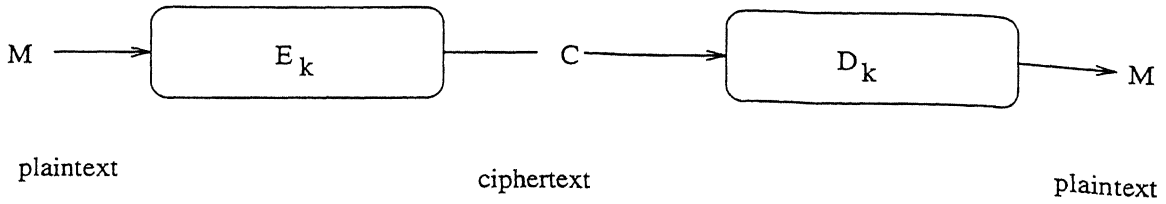


Figure 1.1: A cryptographic system

We begin with an introduction to private and public key cryptography.

1.1.1 Private key cryptography

A cryptographic system has five components:

- A plaintext message space, M.
- A ciphertext message space, C.
- A key space, k.
- A family of enciphering transformations, $E_k : M \rightarrow C$ where k $\in K$.
- A family of deciphering transformations, $D_k : C \rightarrow M$ where k $\in K$.

In a private key cryptosystem, A and B, initially agree upon a secret key, k $\in$ K. For a given k, D_k is the inverse of E_k,
i.e.,

$$D_k(E_k(M)) = M$$

for every plaintext message M.

There are specific requirements for secrecy and authencity in a cryptographic system. **Secrecy** requires that a cryptanalyst not be able to determine plaintext data from intercepted ciphertext.

Data authencity requires that a cryptanalyst not be able to substitute false ciphertext C' for a ciphertext without detection.

Cryptosystems must satisfy three general requirements:

- The enciphering and deciphering transformations must be efficient for all keys.
- The system must be easy to use.
- The security of the sysyem should depend only on the secrecy of the keys and not on the secrecy of the enciphering and deciphering algorithms, i.e., knowing E_k or D_k need not reveal k.

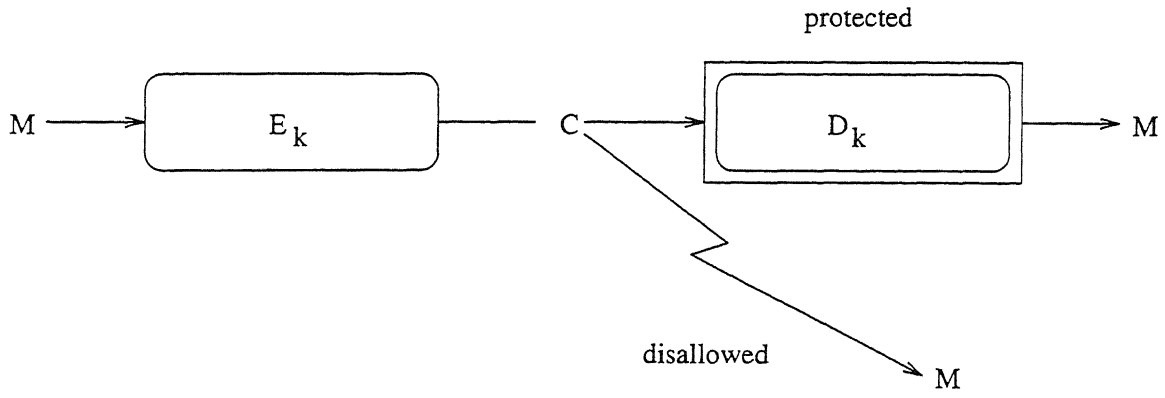


Figure 1.2: Secrecy

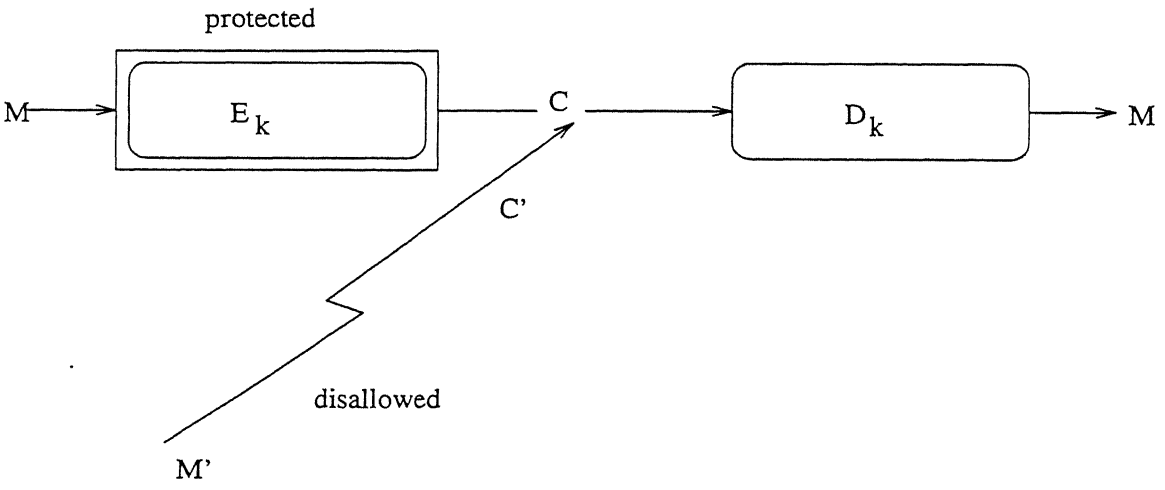


Figure 1.3: Authencity

The most widely used private key cryptosystem today is Data Encryption Standard (DES). It was developed by IBM and subsequently adopted as U.S. standard in 1977 by National Bureau of Standards for the protection of unclassified data. Keys in DES are only 56 bits in length.

Although private key cryptography is adequate for many applications, it has the following disadvantages:

- *Key distribution problem*: The two users have to select a key in secret before they can commence communication over an insecure channel. A secret channel for selecting a key may not be available.

- *Key management problem*: In a network of 'n' users, every pair of users must share a secret key ,for a total of n(n-1)/2 keys.If 'n' is large then the number of keys becomes unmanageable.

- *No signatures possible*: A digital signature allows the receiver of a message to convince any third party that the message infact originated from the sender. In a private key cryptosystem, A and B, have the same capabilities for encryption and decryption and thus B cannot convince a third party that message he received from A infact originated from A.

Diffie and Hellman invented Public key cryptography (PKC) in 1976 to address these deficiencies [7].

1.1.2 Diffie-Hellman Key Exchange

In [7] Diffie and Hellman described the protocol known as*Diffie-Hellman key exchange* in terms of an arbitrary group.

- (setup) A and B publicly select a finite group G and an element $\alpha \in G$.

- A generates a random integer a , computes α^a in G and transmits α^a to B over a public communications channel.

- B generates a random integer b , computes α^b in G and transmits α^b to A over the same channel.
- A receives α^b and computes $(\alpha^b)^a$.
- B receives α^a and computes $(\alpha^a)^b$.

A and B now share the common group element α^{ab}. Note that an eavesdropper C knows G, α, α^a, and α^b, and his task is to use this information to reconstruct α^{ab}. This problem is commonly called **Diffie-Hellman problem**.

The problem of computing a , given$G$, α and α^a is called the **discrete logarithm problem**. The security of the Diffie-Hellman key exchange is based on the intractibility of the discrete logarithm problem.

1.1.3 Public Key Cryptography

The requirements of a public key system differ strikingly from those of conventional cryptographic systems. What Diffie and Hellman proposed is to use an encryption algorithm, E, and a decryption algorithm, D, with E and D chosen so that deriving D even given a complete description of E would be effectively impossible.

Now we establish a secure channel between A and B who never had any previous contact. Each user A has a public enciphering transformation E_A described by a *public key*, which may be registered with a public directory , and a private deciphering tranformation D_A described by a *private key* which is known only to that user.

Here secrecy and authencity are provided by the seperate transformations. Suppose user A wishes to send a message M to another user B. If A knows B's public transformation E_B, A can transmit M to B in secrecy by sending the ciphertext $C = E_B(\mathrm{M})$. On receipt, B deciphers C using B's private transformation D_B, getting

$$\begin{aligned} D_B(C) &= D_B(E_B(M) \\ &= M. \end{aligned}$$

To acheive both secrecy and authencity, the sender and receiver must each apply two sets of transformations. First A's private transformation D_A is applied. Then A enciphers the result using B's public enciphering transformation E_B, and transmits the doubly transformed message

$$C = E_B(D_A(M))$$

to B.

B recovers M by first applying B's own private deciphering transformation D_B, and then applying A's public transformation E_A to validate its authencity, getting

$$\begin{aligned} E_A(D_B(C)) &= E_A(D_B(E_B(D_A(M)))) \\ &= E_A(D_A(M)) \\ &= M \end{aligned}$$

To use a public-key system, the ciphertext space C must be equivalent to the plaintext space M so that any pair of transformations E_A and D_A can operate on both plaintext and ciphertext messages. Furthermore, both E_A and D_A must be mutual inverses so that

$$\begin{aligned} E_A(D_A(M)) &= D_A(E_A(M)) \\ &= M \end{aligned}$$

Observe the following :

- there is no longer the need to exchange keys in secret prior to communicating.
- there is only one key pair associated with each user.

Public key cryptosystems thus overcome the key distribution and key management problems inherent in private key systems.

1.1.4 Digital signatures

A *digital signature* [22] is a property private to a user or a process that is used for signing messages. Let B be the recipient of a message M signed by A. Then A's signature must be able to satisfy these requirements.

- B must be able to validate A's signature on M.
- It must be impossible for anyone, including B, to forge A's signature.
- In case A should deny signing a message M, it must be possible for a judge or a third party to resolve a dispute arising between A and B.

A *digital signature*, therefore establishes sender's authencity; it is analogous to ordinary written signature.

Public-key authentication systems provide a simple scheme for implementing digital signatures.

1.1.5 RSA Cryptosystem

The RSA cryptosystem was invented in 1977 by Rivest, Shamir and Adleman and was the first realization of Diffie and Hellman's abstract model for PKC. The enciphering and deciphering tranformations are based on Euler's generalisation of *Fermat's theorem*. Let us look into some definitions before discussing the RSA scheme.

Definition 1 *Given an integer n, $\phi(n)$ is the number of elements in the reduced set of residues modulo n, i.e., $\phi(n)$ is the number of positive integers less than n that are relatively prime to n.*

In general, for an arbitrary n, ϕ(n) is given by

$$\phi(n) = \prod_{i=1}^{t} p_i^{e_i - 1}(p_i - 1)$$

where

$$n = p_1^{e_1} p_2^{e_2} ... p_i^{e_i}$$

Fermat's Little Theorem 1 *Let n be a prime. Then for every a such that gcd(a.n) = 1,*

$$a^{n-1} \equiv 1 \pmod{n}$$

Euler provided a generalised version of the above theorem as follows.

Euler's generalisation 1 *For every a and n such that gcd(a,n) = 1,*

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

To set up a RSA system, each user A picks two large primes p and q and computes their product

$$n = pq$$

The group used is $G = Z_n^*$, the multiplicative group of units in the integers modulo n. It is well known that the order of G is

$$\phi(n) = (p\text{-}1)(q\text{-}1)$$

where ϕ denotes the Euler phi function. The primes p and q are very large. Although n is public, the factors p and q are kept secret. The decryption key, integer d, is chosen to be a large random integer which is relatively prime to $\phi(n)$.

$$\gcd(d,\phi(n)) = 1$$

The encryption key, integer e, is the multiplicative inverse of d, modulo $\phi(n)$.

$$ed = 1 \bmod \phi(n)$$

The message is represented as an integer between 0 and n-1. The message is encrypted by raising it to the e^{th} power modulo n. To decrypt, the ciphertext is raised to another power d modulo n.

$C = \mathrm{E}(M) = M^e \pmod{n}$, for a message M.

$M = \mathrm{D}(C) = C^d \pmod{n}$, for a ciphertext C.

Since $\phi(n)$ cannot be determined without knowing the prime factors p and q, it is possible to keep d secret even if e and n are made public. Thus, the security of RSA scheme depends upon the difficulty of factoring n into p and q.

1.1.6 ElGamal Cryptosystem

Let G be a finite group of order n and assume that the discrete logarithm problem in G is intractable. In 1985, T.ElGamal proposed the following public key scheme based on discrete exponentiation.

- (setup) A finite group G and element $\alpha \in G$ are chosen. Each user picks a random integer l (the private key) and makes public α^l (the public key). We suppose that messages are elements of G and that user A wishes to send a message M to user B.

- A generates a random integer k and computes α^k

- A looks up B's public key α^b and computes $(\alpha^b)^k$ and $M\alpha^{bk}$.

- A sends to B the pair of group elements $(\alpha^k, M\alpha^{bk})$.

- B computes $(\alpha^k)^b$ and uses this to recover M.

The security of the Elgamal Cryptosystem is based on the difficulty of the discrete logarithm problem.

1.2 Purpose of the present work

In a finite abelian group if an element is obtained as a multiple of another known element (the "base"), the discrete logarithm problem consists in finding the integer that is multiplied by the base to get the element. Whenever we have a finite abelian group for which the discrete logarithm problem appears to be intractable, we can construct various public key cryptosystems in which taking large multiples of a group element is the trapdoor function. Such cryptosystems were first constructed from the multiplicative group of a finite field. However, because certain special techniques are available for attacking the discrete logarithm problem in that case (especially when the field has characteristic two, see [18]), it is worthwhile to study other sources of finite abelian groups.

It has already been described in [11] how the group of points on an elliptic curve can be used to construct public key cryptosystems.The purpose of the present work is to discuss **a more general class of groups** obtained from the jacobians of hyperelliptic curves [3, 12].These jacobian varieties seem to be a rich source of

finite abelian groups for which so far as is known ,the discrete logarithm problem is intractable.

Hyperelliptic cryptosystems potentially provide equivalent security as the existing public key schemes, but with shorter key lengths. Having short key lengths means smaller bandwidth and memory requirements and can be a crucial factor in some applications, for example the design of smart card systems.

Another advantage to be gained is that each user may select a different curve C, even though all users use the same underlying field K. Consequently, all users require the same hardware for performing the field arithmetic, and the curve C can be changed periodically for extra security.

We pay special attention to the case when the ground field has characteristic two, because arithmetic over such fields is particularly amenable to efficient implementation and because it is in that case that the multiplicative group of the field does not provide secure cryptosystems unless the size of the field is extremely large.

1.3 Organization of thesis

The thesis is organized as follows:

- Chapter 2 introduces the preliminary aspects of algebraic geometry which are required in the work. Acquaintance is made with the notion of rational functions and divisors.

- Chapter 3 introduces the hyperelliptic curves and the cryptosystems. The problem of determining the number of F_{q^n} points on the jacobian for varying n is discussed. The concept of normal basis is introduced. We describe how such public key cryptosystems as the Diffie-Hellman key exchange and ElGamal scheme can be carried over to these jacobians.

- Chapter 4 gives a detailed account of various implementations done as part of this thesis. Certain curves have been found which give a large prime factor in

the order of their jacobian. Diffie-Hellman key exchange and ElGamal scheme is carried over these curves. Results have been tabulated wherever required.

Chapter 2

FEW PRELIMINARIES

This chapter contains certain aspects of algebraic geometry that we need in our work.

2.1 Homogeneous Polynomials and Projective Space

Suppose $f(x_1, ..., x_n)$ in $Q[x_1, ..., x_n]$, (Q denotes the rational numbers), is a homogeneous polynomial of degree m, i.e.,

$$f(tx_1, ..., tx_n) = t^m f(x_1, ..., x_n).$$

A point $(a_1, ..., a_n)$ of C^n satisfies

$$f(x_1, ..., x_n) = 0 \tag{2.1}$$

if and only if $(ta_1, ..., ta_n)$ does, for all t in C. Of course, $\mathbf{0}$=(0,...,0) always satisfies (2.1) and is called its *trivial solution.* This leads to the concept of a projective space, see[1, 4, 9].

For any field K, let

$$K^n = \{(x_1, ..., x_n) \mid x_j \in K, j = 1, ..., n\}$$

be the vector space of n-tuples of elements of K. On the set

$$K^{n+1} - \{0\} = \{\mathbf{x} \in K^{n+1} \mid \mathbf{x} \neq 0\}$$

define an equivalence relation by $\mathbf{x} \sim \mathbf{y}$ if and only if $\mathbf{y} = t\mathbf{x}$ for some t in K^*, (K^* is the group of units of a ring K). Then the set

$$P^n(K) = \{K^{n+1} - \{0\}\}$$

of equivalence classes is called the *projective space over* K, i.e., $P^n(K)$ consists of all nonzero vectors in K^{n+1} with two vectors $\mathbf{x}$, $\mathbf{y}$ not considered different if $\mathbf{y} = t\mathbf{x}$ with t in K^*. We can also think of $P^n(K)$ as consisting of lines through the origin 0 of K^{n+1}. The *affine space* $A^n(K) = K^n$ may be regarded as a subset of $P^n = (K)$ by the inclusion map

$$A^n(K) \ni (x_1, ..., x_n) \rightarrow (x_1, ..., x_n, 1) \in P^n(K).$$

2.2 Plane Algebraic Curves

A *curve C defined over* Q is the set of solutions of a polynomial equation

$$f(x, y) = 0 \tag{2.2}$$

with $f(x,y) \in Q$[x,y]. Suppose that

$$f(x, y) = \prod_{j=1}^{r} p_j(x, y)^{m_j}$$

is the factorization of $f(x,y)$ into irreducible factors. Then the curves C_j defined by

$$p_j(x, y) = 0, \ j = 1, ..., r$$

are all irreducible. The curve C is the union of $C_1, ..., C_r$, called the *(irreducible)componen* of C, see [1, 4].

A *projective curve over* Q is the set of solutions of a homogeneous equation

$$F(X, Y, Z) = 0 \tag{2.3}$$

with $F(X,Y,Z)$ in $Q[X, Y, Z]$.

One can *homogenize* (2.2) to get an equation like (2.3) by putting $x = X/Z, y = Y/Z$ and then multiplying throughout by $Z^{deg(f)}$. Now that we have obtained (2.3)

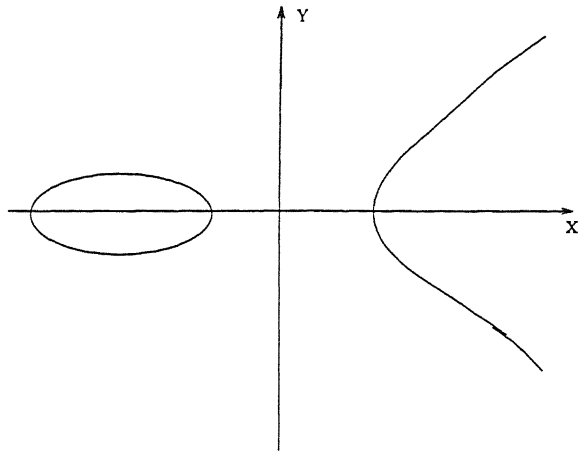


Figure 2.1: Elliptic curve in case of three real roots

from (2.2) in this manner, we say that (2.3) is a *projective model* for the *affine curve* C defined by (2.2). The solutions of (2.2) are precisely those solutions of (2.3) for which $Z \neq 0$,because(a,b) is a point on (2.2) if and only if $(a,b,1)$ is a point on (2.3).

A point(a,b,c) on (2.3) with $c = 0$ is called a ıpoint at infinity on C. The *line at infinity* is the set of all points (X,Y,Z) in $P^2(C)$ for which $Z = 0$.

The points on (2.2) are called the *affine part* of the projective curve(2.3). A projective curve is *complete* in the sense that it has the points at infinity that are missing from its affine part.

We illustrate the concept through an example:

An *elliptic curve* E, Fig 2.1, *defined over* a subfield k of C and written as E/k is a projective curve given by

$$Y^2Z = X^3 + AXZ^2 + BZ^3(A, B \in k),$$

where the quantity Δ = $-4A^3 - 27B^2$, called the *discriminant* of E, is nonzero.

There is only one point at infinity on the above projective curve. It is given by$Z = 0$ and is O = (0,1,0). It is regarded as a rational point and we may think of E as the affine curve

$$y^2 = x^3 + Ax + B,$$

together with the point O = (0,1,0) at infinity.

2.3 Singularities of a curve

Let C be a projective curve defined by(2.3). We say that C is of *order* n if $\deg(F) = n$. We call C *linear, quadratic, cubic, quartic or quintic* according as $n = 1, 2, 3, 4,$ or 5. A point P on(2.3) is a *multiple point* of *multiplicity* $r \geq 1$ or an *r-fold* point if all the partial derivatives of $F(X, Y, Z)$ of order $< r$ vanish at P, but there is a partial derivative of order r that does not vanish at P.

If r=1, P is called a *regular* or a *nonsingular*point. (For a point$P$, not at infinity, to be a regular point is equivalent to at least one of dy/dx, dx/dy being well defined, i.e., to the curve having a unique tangent at P.) If $r > 1$, P is called a *singular* point or a *singularity* of C. We say that P is a *double* or a *triple* point according as $r = 2$ or 3. A curve is *singular* if it has a singularity, otherwise it is *nonsingular* or *smooth*.

Example : Let C be the affine curve defined by $y^2 = x^3$. To homogenize this equation, we put $x = X/Z$, $y = Y/Z$ and obtain

$$F(X,Y,Z) = Y^2Z - X^3 = 0.$$

The singularities of C must satisfy

$$\partial F/\partial X = -3X^2 = 0,$$

$$\partial F/\partial Y = 2YZ = 0,$$

$$\partial F/\partial Z = Y^2 = 0,$$

which are therefore of the form$(0,0,Z)$ with $Z \neq 0$. Thus $(0,0,1)$ is the only singularity of C. In fact, it is a double point, because

$$\partial^2 F/\partial Y^2 \,|_{(0,0,1)} \neq 0.$$

2.4 Birational Geometry

Suppose K is a field and C is an irreducible curve given by a polynomial equation

$$f(x,y) = 0$$

with coefficients in K. If L is another field with $K \subseteq L$, a point $P = (x,y)$ on C is called an *L-rational point* if $x, y \in L$.

Our goal is to study the rational points on curves defined over Q, which we may assume to be irreducible over Q. The problem gets more complicated as the degree(or rather the genus, a term to be explained later,) of the curve increases. We start with the simplest curves. Unless stated otherwise, we shall assume that $K = L = Q$. Consider the field of rational functions

$Q(x_1, ..., x_n)$

$= \phi(x_1, ..., x_n) = \frac{f(x_1,...,x_n)}{g(x_1,...,x_n)} \mid f, g \in Z[x_1, ..., x_n], g \neq 0.$

It is the quotient field of the ring of polynomials $Q[x_1, ..., x_n]$. *If* $\phi = f/g \in Q($ $x_1, ..., x_n)$ and $P = (a_1, ..., a_n)$ is a rational point, i.e., if all $a_i \in Q$, then $\phi(P) \in Q$ [provided $g(P) \neq 0$].

Def. An irreducible curve C defined by $f(x,y) = 0$ is called a *rational curve* if there are rational functions $\phi_1(t), \phi_2(t)$ in $Q(t)$, such that

- for almost all t, f$(\phi_1(t), \phi_2(t)) = 0$; and
- for almost all P on C, $P = (\phi_1(t), \phi_2(t))$.

We say that C is *parametrized* by t.

Example: For the cubic, Fig 2.2,

$$y^2 = x^2(x+1)$$

$$x = \phi_1(t) = t^2 - 1,$$

$$y = \phi_2(t) = t(t^2 - 1).$$

Almost all the rational points on these curves are obtained by taking t from Q.

Let C_1, C_2 be two curves such that the coordinates of almost all points on C_2 are rational functions of the coordinates of points on C_1 and vice versa. This means that there are rational functions

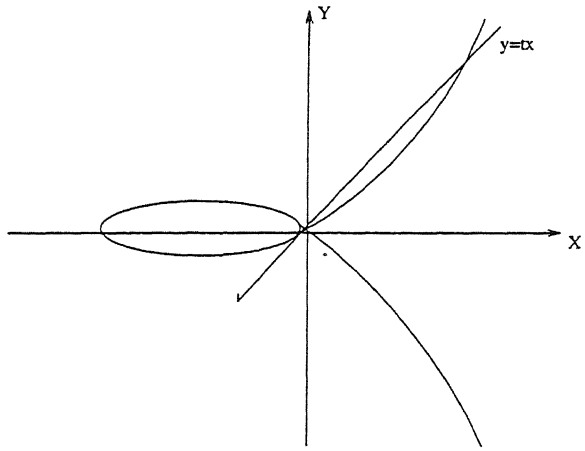


Figure 2.2: Parametrizing a singular cubic

$$\phi_j(x,y) = \frac{f_j(x,y)}{g_j(x,y)} \in Q(x,y),\ j = 1,2$$

such that

- for almost all P on C_1, $g_1(P)g_2(P) \neq 0$; and
- the set $\{\phi(P) = (\phi_1(P), \phi_2(P)) \mid P \in C_1, g_1(P)g_2(P) \neq 0\}$

accounts for almost all points on C_2.

A similar statement holds in the other direction. Two such curves C_1 and C_2 are called *birationally equivalent* . If C_1 and C_2 are birationally equivalent,then almost all points on one curve are obtained from those on the other by the *rational maps* :

$$\phi : C_1 \to C_2 \tag{2.4}$$

$$\psi : C_2 \to C_1 \tag{2.5}$$

given by

$$\phi(P) = (\phi_1(P), \phi_2(P))$$

and

$$\psi(Q) = (\psi_1(Q), \psi_2(Q)).$$

If $\psi\phi : C_1 \to C_1$ and $\phi\psi : C_2 \to C_2$ are the identity functions, then (2.4) and (2.5) describe a birational correspondence between C_1 and C_2.

Example: Let C_1 be the Fermat curve given by

$$x^3 + y^3 = 1.$$

If we make the rational substitution

$$x = \frac{6}{X} + \frac{Y}{6X}, y = \frac{6}{X} - \frac{Y}{6X},$$

$\leftrightarrow$

$$X = \frac{12}{x+y}, Y = 36\frac{x-y}{x+y},$$

we get the curve C_2 with equation

$$Y^2 = X^3 - 432$$

Remark Birational equivalence between curves is an equivalence relation. If C_1, C_2 are birationally equivalent, then C_1 has infinitely many rational points on it if and only if C_2 does.

2.5 The Genus of a curve

As expected, the curves get more complicated to study as their degree increases. But it is really the genus, [4, 9], that tells how complicated a curve is at least as far as the study of its rational solutions is concerned.

Theorem:*Let C_1 be an irreducible curve of order n with m double points as its only singularities. Then*

$$g = g(C_1) = \frac{(n-1)(n-2)}{2} - m$$

is a non-negative integer.

Theorem:*Let C be an irreducible curve that is birationally equivalent to two curves C_1 and C_2 each with double points as its only singularities. Then $g(C_1) = g(C_2)$.*

The common value $g(C) = g(C_1) = g(C_2)$ is called the *genus* of the (irreducible) curve C. Note that g is a non-negative integer.

The genus of a curve is invariant under birational equivalence.

Example: Let L be a line. We have $n = 1$ and $m = 0$. Therefore,

$$g(L) = 0.$$

The *elliptic curve* defined earlier has genus one.

Any curve of genus larger than one has only finitely many rational points. A curve of genus one may or may not have infinitely many rational solutions and it is still an open question how to decide whether or not such a curve has only finitely many solutions.

2.6 Divisor Theory

Divisors, [1, 17, 25, 26] are useful devices for keeping track of the zeros and poles of a rational function.

Let X be a smooth projective curve over K. A *divisor* D on X is a finite formal sum

$$D = \sum a_P.P, P \in X, a_P \in Z.$$

The *support* of a divisor D, denoted *supp(D)*, is the set of points $\{P \in X \mid a_P \neq 0\}$.

The set of all divisors, denoted by $\mathbf{D}$, forms a group, where the addition is given by

$$\sum_{P \in X} a_P(P) + \sum_{P \in X} b_P(P) = \sum_{P \in X}(a_P + b_P)(P).$$

$\mathbf{D}$ is the free abelian group generated by the points of X.

The *degree* of a divisor $D = \sum a_P(P)$ is the integer $\deg(D) = \sum a_P$. The degree map $\mathbf{D} \rightarrow Z$ is surjective, its kernel is denoted by D^0. Therefore D^0 is the set of all divisors of degree 0 and it is a subgroup of $\mathbf{D}$.

If $a_P \geq 0$ for any P then we call $D = \sum a_P(P)$ an *effective divisor* and write $D \geq 0$. If moreover $D \neq 0$ we call it *positive*. The set of effective divisors is denoted by $\mathbf{D}^+$.

Given two divisors $D_1 = \sum m_i P_i$ and $D_2 = \sum n_i P_i$ in $\mathbf{D^0}$, we define gcd (D_1, D_2) to be $(\sum min(m_i, n_i) P_i)^0$.

If E is a smooth, irreducible affine curve defined by

$$r(x, y) = 0,$$

where $r \in K[x, y]$, then the *coordinate ring* of over K, denoted $K[E]$ is the integral domain

$$K[E] = K[x,y]/(r),$$

where (r) denotes the ideal in $K[x,y]$ generated by r. Similarly, we define

$$\overline{K}[E] = \overline{K}[x, y]/(r).$$

Observe that for each $l \in \overline{K}[E]$ we can repeatedly replace any occurrence of y^2 by $y^2 - r(x, y)$ to finally obtain a representation

$$l(x,y) = v(x)+yw(x), \text{ where } v(x), w(x) \in \overline{K}[x].$$

The *function field* $K(E)$ of E over K is the field of fractions of $K[E]$. (We know that if I is an integral domain then its field of fractions F is the set of equivalence classes of quotients a/b, $a, b \in I$, $b \neq 0$, where we identify $a_1/b_1, a_2/b_2 \in F$ if $a_1 b_2 = a_2 b_1$. Addition and multiplication in F are defined in the natural way.) Similarly, $\overline{K}(E)$, the function field of E over $\overline{K}$, is the field of fractions of $\overline{K}[E]$. The elements of $\overline{K}(E)$ are called *rational functions.*

Let $f \in \overline{K}(E)^*$ be a non-zero rational function and $P \in E \mid \{O\}$. Then f is said to be *defined at* P if there exists a representation $f = g/h$, $g, h \in \overline{K}[E]$, with $h(P) \neq 0$. If f is defined at P, we put $f(P) = g(P)/h(P)$. It is easy to see that it is well-defined, i.e., the value $f(P)$ does not depend on the choice of g and h. If $f(P) = 0$, then f is said to have a *zero* at P. If f is not defined at P then f is said to have a *pole* at P, in which case we write $f(P) = \infty$.

Example: Consider the elliptic curve E: $y^2 = x^3 - x$ over a finite field $K = F_q$, with char$(K) \neq 2, 3$. Let $P = (1,0) \in E$, and let $f = (x^2 - x)/y \in \overline{K}(E)$.

Note that if f is considered as a quotient of polynomials, i.e., $f \in \overline{K}(x, y)$, then f is undefined at P. However, as an element of $\overline{K}(E)$,

$$f = \frac{x^2 - x}{y} = \frac{(x^2 - x)y}{y^2} = \frac{(x^2 - x)y}{x^3 - x} = \frac{y}{x + 1},$$

whence $f(P) = 0$.

In defining the value of f at the point O the following approach is taken. For $l \in \overline{K}[E]$ we can write $l(x,y) = v(x)+yw(x)$, where $v(x)$, $w(x) \in \overline{K}[x]$. Assign a weight of 2 to x and a weight of $(2 \cdot g+1)$to y. We define the *Degree* of l by

$$\text{Deg}(l) = \max(2deg_x(v),\ 2.\text{genus}+1+2deg_x(w)).$$

Now, let $f = g/h$, where $g, h \in \overline{K}[x, y]/(r)$.

- If Deg(g)<Deg(h), then $f(O) = 0$.
- If Deg(g)>Deg(h), then $f(O) = \infty$.
- If Deg(g) = Deg(h), then if the highest degree terms in g and h are ax^d and bx^d respectively then $f(O) = a/b$. Otherwise the highest degree terms are cyx^d and dyx^d, in which case $f(O) = c/d$.

Example: Consider the elliptic curve E: $y^2 = x^3 + ax + b$. Let$f = y, g = x/y$, $h = (x^2 - xy)/(1 + xy) \in \overline{K}(E)$. Then $f(O) = \infty$,$g(O)$=0, and $h(O) = -1$.

For each point $P \in E$ there exists a rational function $u \in \overline{K}(E), u(P)$ =0, such that if $f \in \overline{K}(E)^*$ then we can write $f = u^d s$, where $s \in \overline{K}(E), s(P) \neq 0, \infty$. The integer d does not depend on the choice of u. The function u is called a *uniformizing parameter* for P. The next theorem[9] aids in finding the uniformizing parameters.

Theorem:*Let $P \in E$. If l: $ax+by+c=0$ is any line through P that is not the tangent line to E at P, then l is a uniformizing parameter for P.*

Example: Consider the elliptic curve E:$y^2 = x^3 + ax + b$ over a finite field $K = F_q$, char(K)$\neq$2,3.

- Let $P = (c,d)$ not a member of E[2]. The tangent line to E at P is

$$(-3c^2 - a)(x - c) + 2d(y - d) = 0,$$

Since $d \neq 0$, a uniformizing parameter for P is $u = x\text{-}c$.

- Let $P = (c,0) \in E$ be a point of order 2. The tangent line to E at P is

$$(-3c^2 - a)(x - c) = 0.$$

Therefore $u = y$ is a uniformizing parameter for P.

- To find a uniformizing parameter for O we need to work with a different set of coordinates. Recall that the homogeneous equation for E is $Y^2Z = X^3 + aXZ^2 + bZ$ Choosing the affine coordinates $v = X/Y, w = Z/Y$, the equation for E is transformed to $f(v, w) = v^3 + avw^2 + bw^3 - w = 0$. Note that $O = (0,0)$ in (v,w) coordinates. Now,

$$\frac{\partial f}{\partial v}(O) = 0,$$

and

$$\frac{\partial f}{\partial w}(O) = -1,$$

- so the equation of the tangent line to E at O is $w = 0$. The line $v = 0$ passes through O and is not the tangent line at O. Reverting back to the original (x,y) coordinates, $u = x/y$ is a uniformizing parameter for O.

Let $f \in \overline{K}(E), P \in E$. Write $f = u^d s$, where u is any uniformizing parameter for $P, s \in \overline{K}(E)$, and $s(P) \neq 0, \infty$.
The *order of* f *at* P is defined to be d, and we write $ord_P(f) = d$. The point P is a zero of f if and only if $ord_P > 0$, in which case its *multiplicity* is defined to be $ord_P(f)$.

Similarly, the point is a pole of f if and only if $ord_P(f) < 0$, in which case its multiplicity is defined to be $-ord_P(f)$.Since a function f has only a finite number of zeros and poles on E, we can define div(f), the divisor of f , as

$\text{div}(f) = \sum_{P \in E} ord_P(f)(P)$.

A fundamental fact about rational functions is that if $f \in \overline{K}(E)^*$, then $\text{div}(f) \in D^0$. Moreover, $\text{div}(f) = 0$ if and only if $f \in \overline{K}^*$.

Example: Consider the elliptic curve E: $y^2 = x^3 + ax + b$ over a finite field $K = F_q$, $\text{char}(K) \neq 2, 3$.

- Let $P = (c,d)$,(P is not a member of $E[2]$). Then

$$\text{div}(x - c) = (P) + (-P) - 2(O).$$

- Let $P_1, P_2, P_3, \in E$ be points of order 2. Then

$$\text{div}(y) = (P_1) + (P_2) + (P_3) - 3(O).$$

- Assume that $b \neq 0$, and let $P_4 = (0, \sqrt{b}), P_5 = (0, -\sqrt{b})$. Then

$$\text{div}(\tfrac{x}{y}) = (P_4) + (P_5) + (O) - (P_1) - (P_2) - (P_3).$$

A divisor $D \in D^0$ is *principal* if $D = \text{div}(f)$ for some $f \in \overline{K}(E)^*$. The following is a useful characterization of principal divisors.

Theorem:*Let $D = \sum n_P(P)$ be a divisor. Then D is principal if and only if $\sum n_P = 0$ and $\sum n_P(P) = O$.*

Let D_l denote the set of principal divisors. If $f_1, f_2 \in \overline{K}(E)$, then $\text{div}(f_1 f_2) = \text{div}(f_1) + \text{div}(f_2)$;

it follows that D_l forms a subgroup of D^0. The quotient group D^0/D_l is called the *jacobian* **J** of the curve E.

Two divisors $D_1, D_2 \in D^0$ are said to be *equivalent*, denoted $D_1 \sim D_2$, if $D_1 - D_2 \in D_l$, i.e., if $D_1 = D_2 + \text{div}(f)$ for some $f \in \overline{K}(E)$.

Assume we are given a divisor $D \in \mathbf{D}$. We form the set

$$L(D) = f \in \overline{K}(E) \mid (f) + D \succ 0 \cup 0.$$

If $D = \sum_{i=1}^{s} m_i P_i - \sum_{j=1}^{t} n_j Q_j$, where $m_i, n_j > 0$, then $L(D)$ consists of 0 and all functions in $\overline{K}(E)$, that have zeros of order at least n_j at $Q_j, 1 \leq j \leq t$, poles of order at most m_i at $P_i, 1 \leq i \leq s$, and no other poles. $L(D)$ is a vector space over k,[26]. Let the *dimension* of $L(D)$ over k be denoted by $l(D)$.

Lemma:

1 $L(D) = 0$ *if* $\deg(D) < 0$.

2 $L(0) = k$.

3 $L(D)$ *is a finite dimensional vector space over k. If* $deg(D) \geq 0$ *then* $l(D) \leq 1 + deg(D)$.

Now, having being acquainted with the notion of divisors, we introduce the concept of hyperelliptic curves in the next chapter.

Chapter 3

Hyperelliptic cryptosystems

3.1 Introduction

The jacobian of a hyperelliptic curve C defined over a field K forms an abelian group. These jacobian varieties provide a rich class of abelian groups making them attractive for cryptographic implementations. In 1989, Koblitz [12] proposed a variant of discrete log cryptography based on the hyperelliptic discrete log problem (HDLP). The security of the proposed public key cryptosystems is based on the HDLP. These cryptosystems have two potential advantages over systems based on the multiplicative group of a finite field.

- the great diversity of curves available to provide the groups;and
- the absence of subexponential time algorithms (such as those of index calculus type) that could find discrete logs in these groups.

Hyperelliptic curve cryptosystems potentially provide equivalent security as the existing public key schemes, but with shorter key lengths. Having short key lengths means smaller bandwidth and memory requirements.

3.2 Hyperelliptic curves and the Jacobian

Let K be an arbitrary field, and let $\overline{K}$ denote its algebraic closure. We define a *hyperelliptic curve C of genus g* over K to be an equation of the form

$$v^2 + h(u)v = f(u), \tag{3.1}$$

where $h(u)$ is a polynomial of degree at most g and $f(u)$ is a monic polynomial of degree $2g+1$.

Here f and h have coefficients in K, and we require that the curve has no singular points(u,v), i.e., there be no values $u, v \in \overline{K}$ which satisfy (3.1) and also both of the partial derivative equations

$$2v + h(u) = 0$$

and

$$h'(u)v - f'(u) = 0.$$

Let L be a field containing K. By an *L-point* $P \in C$ we mean either the symbol ∞ or else a solution $u = x \in L, v = y \in L$ of (3.1). The latter is called a "finite point" and is denoted by $P_{x,y}$. If σ is an automorphism of L over K, we let P^σ denote $P_{\sigma(x),\sigma(y)}$ and set $\infty^\sigma = \infty$.

We shall denote the group of principal divisors as $\mathbf{P}$. If the divisors of degree 0 are represented as $\mathbf{D}^0$, then the jacobian $\mathbf{J}$ of C is defined to be the quotient group $\mathbf{J} = \mathbf{D}^0/\mathbf{P}$.

If P=(x,y) is a point on the curve C, then so is its "opposite"-defined as $\overline{\mathrm{P}}$=$(x,-y-h(x))$. The points P and $\overline{\mathrm{P}}$ are zeros of the function $(u - x)$, which has a double pole at ∞. Thus the divisor P+($\overline{\mathrm{P}}$) $-2\infty \equiv 0$ (mod $\mathbf{P}$) or $-\overline{\mathrm{P}} \equiv \mathrm{P} - 2\infty$ (mod $\mathbf{P}$). It follows that each element of $\mathbf{J}$ can be represented in the form

$$D = \sum_{i=1}^{r} P_i - r\infty,$$

with the following condition satisfied: If the point $P_i = (x_i, y_i)$ appears in D, then the point $\overline{P_i}$ does not appear as one of the P_j(j $\neq$i). Such a divisor is called "semireduced".

Riemann−Roch said in [9] that each element of **J** can be uniquely represented by such a divisor, subject to the additional restriction that $r \leq g$. Such divisors are called "reduced".

A semireduced divisor $D = \sum m_i P_{x_i,y_i} - (\sum m_i)\infty$ can be uniquely represented as the gcd of two principal divisors of functions of the form $a(u)$ and $b(u) - v$ (i.e., D=gcd$(a(u)), (b(u)-v))$,where

$$a(u) = \prod (u - x_i)^{m_i}$$

and $b(u)$ is the unique polynomial of degree< deg$a$ satisfying

$$b(x_i) = y_i$$

with appropriate multiplicity when the point P_i appears more than once in D. Explicitly, if P_i appears k timed in D, then $b(u)^2 + h(u)b(u) - f(u)$ must be divisible by $a(u)$.

A divisor D represented in the form gcd$(a(u),\ (b(u)-v))$ is abbreviated as D = div(a,b). D is reduced if and only if deg $a \leq g$.

3.2.1 Group Law

An element of our group **J** is an equivalence class of divisors. Every divisor is equivalent to a unique "reduced" divisor,by which we mean a semireduced D=div(a,b) for which deg $a \leq g$.

Consider two divisors D_1=div(a_1, b_1) and D_2 =div$(a_2, b_2) \in$ **J**. Let O be the identity element of **J**. Then, addition is defined as follows:

a. $O + D_1 = D_1$ and $D_1 + O = D_1$.

b. $-O = O$.

c. If $D_1 \neq O$,then $-D_1 = (a_1, -b_1 - h)$.

d. If $D_2 = -D_1$,then $D_1 + D_2 = O$.

e. If $D_1 \neq O, D_2 \neq O, D_2 \neq -D_1$,then we find a divisor $D \sim D_1 + D_2$ through the algorithm stated below.

As in any abelian group, the notation mD denotes D added to itself m times if m is positive, and $-D$ added to itself $\mid m \mid$ times if m is negative, and $0D = O$.

The algorithm for adding divisors $D \in \mathbf{J}$ consists of two stages.

Given D_1 =div(a_1, b_1) and D_2=div(a_2, b_2), we first find a semireduced divisor D=div(a,b) such that $D \sim D_1 + D_2$. Next, we "reduce" D, i.e., we find $a'(u)$ and $b'(u)$ such that deg $a' \leq g$,deg b' <deg a',and $D \sim$div(a',b').

First we describe Extended Euclidean algorithm which is needed in the composition rules.

Extended Euclidean Algorithm : Given the polynomials $r^{(-2)}$ and $r^{(-1)}$ (whose gcd has to be found), define $p^{(-2)} = 0, p^{(-1)} = 1, q^{(-2)} = 1, q^{(-1)} = 0$ and compute $a^{(k)}(x), r^{(k)}(x), p^{(k)}(x), q^{(k)}(x)$ by the iterative procedure

$$\begin{aligned} r^{(k-2)}(x) &= a^{(k)}(x)r^{(k-1)}(x) + r^{(k)}(x) deg r^{(k)} < deg r^{(k-1)} \\ p^{(k)}(x) &= a^{(k)}(x)p^{(k-1)}(x) + p^{(k-2)}(x) \\ q^{(k)}(x) &= a^{(k)}(x)q^{(k-1)}(x) + q^{(k-2)}(x) \end{aligned}$$

Also,$r^{(-1)}(x)p^{(k-1)}(x) - r^{(-2)}(x)q^{(k-1)} = (-1)^k r^{(k-1)}(x)$, for k = 0,1,2,.... The algorithm will eventually terminate with $r^{(s)} = 0$.Then

$$\begin{aligned} p^{(s)}(x)r^{(s-1)}(x) &= r^{(-2)}(x) \\ q^{(s)}(x)r^{(s-1)}(x) &= r^{(-1)}(x), \\ r^{(s-1)}(x) &= gcd(r^{(-2)}(x), r^{(-1)}(x)) \end{aligned}$$

We wish to find the sum of div(a_1, b_1) and div(a_2, b_2) on the jacobian of the curve $v^2 + hv = f$, where a_1, a_2, b_1, b_2, h, and f are all polynomials in u. (Here h and f have coefficients in K, and a_1, a_2, b_1, b_2 may have coefficients in an extension field of K).

Stage 1

Addition. Let d=$d(u)$ be the gcd of the three polynomials $a_1(u), a_2(u)$, and

$b_1(u) + b_2(u) + h(u)$; and choose $s_1(u), s_2(u)$, and $s_3(u)$ to be polynomials in u such that

$$d = s_1 a_1 + s_2 a_2 + s_3(b_1 + b_2 + h).$$

For this the extended euclidean algorithm is to be used twice.

Set

$$a = a_1 a_2 / d^2$$

and

$$b = (s_1 a_1 b_2 + s_2 a_2 b_1 + s_3(b_1 b_2 + f))/d \pmod{a}.$$

The correctness of the algorithm has been proved in [3].

Thus div(a,b) is semireduced and represents the divisor sum in the jacobian.

Special cases.

1. If a_1 and a_2 have no common factor, then d=1, we can take s_3=0, and so $a = a_1 a_2, b = s_1 a_1 b_2 + s_2 a_2 b_1 \pmod{a}$.

2. When $a_2 = a_1$ and $b_2 = b_1$(, i.e., we are doubling an element of **J**), we can take s_2=0. Assume char K=2 and $h(u)$=1.Then d=1,$s_1 = s_2$=0,s_3 =1,and $a = a_1^2, b = b_1^2 + f \pmod{a}$.

Stage 2

Reduction : Given D=div(a,b) with deg a>g, the following procedure replaces D with an equivalent divisor D'=div(a',b') for which deg a'<deg a. By successively applying the procedure,we eventually obtain D''=div(a'',b'') for which D'' and deg $a'' \leq g$.

We set

$$a' = (f - hb - b^2)/a$$

and then

$$b' = -h - b \pmod{a'}.$$

Remark. In the case g=1(elliptic curves),the reduced divisors D=div$(u = x, y)$ are in one-to-one correspondence with the points $P_{x,y} \in C$. The above algorithm then reduces to the usual formulas for the addition of points on an elliptic curve(see [17]).

Examples: Let K=$\mathbf{F}_2$ be the field of two elements. For $P \in C$ we let $P^{(j)}$ denote P^{σ^j}, where σ^j is the automorphism of $\overline{\mathbf{F}}_2$ given by $x \mapsto x^{2^j}$. In certain cases there are simple formulas expressing $2^k P$ in terms of $P^{(j)}$ which give a short-cut in computing multiples of $D = \sum m_i P_i - (\sum m_i)\infty$. The formulas below all follow by repeated application of Stage 1(special case 2) and then Stage 2 of the algorithm.

1. C is given by $v^2 + v = u^3$, $g = 1$, then

$$2P = -P^{(2)}.$$

2. C is given by $v^2 + v = u^3 + u + 1$, $g = 1$, then

$$4P = -P^{(4)}.$$

3. C is given by $v^2 + uv = u^3 + 1$, $g = 1$, then

$$16P = P^{(4)} - P^{(8)}.$$

A

4. C is given by $v^2 + v = u^5 + u^3$, $g = 2$, then

$$64P = -P^{(12)}.$$

5. C is given by $v^2 + v = u^5 + u^3 + u$, $g = 2$, then

$$8P = P^{(6)}.$$

In the next chapter, we give a general method for reducing the calculation of mP for m large to the computation of linear combinations of $P^{(j)}$ with small coefficients.

3.3 Security aspects of hyperelliptic curves

In terms of hyperelliptic curves, the discrete logarithm problem is the following.

Definition: *Let* $P \in \mathbf{J}$ *over* $\mathbf{F_q}$ *be an element of maximum order* n_1, *and let* $R \in \mathbf{J}$. *Given* P *and* R, *determine the unique integer* $l, 0 < l \leq n_1 - 1$, *such that* $R = lP$, *provided that such an integer exists.*

In order to use the jacobians for cryptosystems, the discrete logarithm problem should be intractable. As for the intractability of the discrete logarithm of the elliptic curve, it has been known that some specific elliptic discrete logarithms are as tractable as multiplicative discrete logarithms (MOV reduction attack, see [27]). Thus it remained open whether these results about elliptic curves could be extended to hyperelliptic curves. The hyperelliptic discrete logarithm is characterized as more intractable than the multiplicative discrete logarithm, see [24]. A problem corresponding the hyperelliptic discrete logarithm is in NP $\cap$ co$-$AM, while a problem corresponding the multiplicative discrete logarithm is in NP $\cap$ co$-$NP. However it remained open whether the problem corresponding to the hyperelliptic discrete logarithm is in NP $\cap$ co$-$NP.

Okamoto and Sakurai in [19] extended the above results of elliptic curves to derive similar results for hyperelliptic curves.

The Weil pairing was originally defined over elliptic curves by A.Weil in [25]. Lang generalized the Weil pairing over the abelian varieties. Okamoto and Sakurai defined the extended Weil pairing on the jacobian of the hyperelliptic curves, which is a specific class of Lang's generalization.

Definition: *Let* C *be a hyperelliptic curve defined over* $\overline{K}$, *and* $\mathbf{J}$ *be the jacobian on* C ($\mathbf{J} = \mathbf{D^0/P}$). *Let* $\mathbf{J}[m]$ *be* $\mathbf{D^0}[m]/\mathbf{P}$ *and* μ_m *be the set of* m*-th roots of unity, where* $\mathbf{D^0}[m] = (D \mid D \in \mathbf{D^0}$ *and* $mD = O)$ *and the characteristic of* K *is prime to* m.

Then, we deine a pairing (the extended Weil pairing) as a mapping

$$e_m : \mathbf{J}[m] * \mathbf{J}[m] \rightarrow \mu_m.$$

The algorithm for computing this pairing has been given in [19].These results formed the basis of the reduction of the HDLP to the conventional multiplicative discrete

logarithms,which is an extension of the result by MOV.

Frey and Ruck Reduction Attack :

If the order of the jacobian contains a prime factor of r decimal digits, then for security purpose r should not divide $q^k - 1$ for small values of k for which the discrete logarithm in F_{q^k} is feasible. In general, $1 \leq k \leq 2000/log_2 q$ suffices.

The most powerful general algorithm, for solving discrete logarithm problem,known at present is the baby-step giant-step algorithm of Shanks [17]. Let G be a group of order n and consider the interval $I(n)$ of integers from 0 to $n-1$. Let α and β be two members of G, and suppose we want to determine (if such exists), an integer x in $I(n)$ such that $\alpha^x = \beta$. The algorithm is as follows:

Let $m=\lceil \sqrt{n} \rceil$. Then precompute a list of pairs (i, α^i) for $0 \leq i \leq m$. Then for each $j, 0 \leq j \leq m$, compute $\beta\alpha^{-jm}$, and see if this element is the second component of a member of the precomputed list. If $\beta\alpha - jm = \alpha^i$ for some i, $0 \leq i \leq m$, then $\beta = \alpha^{i+jm}$, otherwise no solution exists for x.

This algorithm requires a table with $O(m)$ entries. To sort the table and search it for each value of j requires in total $O(m\log m)$ operations. A group of approximately 10^{40} elements would render this attack infeasible with current technology. Pohlig Hellman method takes advantage of the factorization of the order n of the group.

$$n = \prod_{i=1}^{t} p_i^{\lambda_i}$$

If $x = log_\alpha \beta$ then the approach is to determine x modulo $p_i^{\lambda_i}$ for each i, and then use the Chinese remainder theorem to compute x modulo n. This method fails if the order n contain a prime factor of about 40 decimal digits.

Therefore a hyperelliptic curve selected for cryptosystem should have :

1. A prime factor of about 40 decimal digits in the order of its jacobian
2. should not be mapped into an extension of finite field for small values of k

3.4 Order of Jacobian

We assume that K=$\mathbf{F}_q$ is a finite field with q elements.The abelian group $\mathbf{J}(L)$ is finite for any finite extension L=$\mathbf{F}_{q^n}$.We set

$$N_n = \#(\mathbf{J}(\mathbf{F}_{q^n})).$$

Koblitz in [12] gave a method for computing the order through zeta fuction. A basic fact about the N_n is that there is a simple method for determining the sequence $N_1, N_2, ...$ by counting the number of $\mathbf{F}_{q^n}$ solutions of the equation of C fot the first g values n=1,...,g.We now describe this method.

Let $M_n = \#(C(\mathbf{F}_{q^n})) - q^n$, where $\#(C(\mathbf{F}_{q^n}))$ is the number of solutions u,v $\in\mathbf{F}_{q^n}$ of the equation $v^2 + h(u)v = f(u)$.Associated with the curve $C$is a polynomial $Z(T)$ of degree $2g$ with integer coefficients having the form

$$\begin{aligned} Z(T) &= T^{2g} + a_1T^{2g-1} + ... + a_{g-1}T^{g+1} + a_gT^g \\ &\quad + qa_{g-1}T^{g-1} + q^2a_{g-2}T^{g-2} + ... + q^{g-1}a_1T + q^g \\ &= \prod_{j=1}^{g}((T-\alpha_j)(T-\overline{\alpha}_j)), \end{aligned} \tag{3.2}$$

where $a_1, ..., a_g \in Z$ and where $\overline{\alpha}_j = q/\alpha_j$ (i.e., the roots are complex numbers of absolute value $\sqrt{q}$).The relationship between $Z(T)$ and M_n is the following power series identity (where $\overline{Z}(T)$ denotes the reciprocal polynomial $T^{2g}Z(1/T)$):

$$\log(\overline{Z}(T)) = \textstyle\sum_{n=1}^{\infty} \frac{M_n}{n} T^n.$$

Thus the first g values $M_1, ..., M_g$ are enough to determine the coefficients of $Z(T)$.

Once $Z(T)$ has been found,N_n can be determined from the formula

$$N_n = \prod_{j=1}^{g} \mid 1 - \alpha_j^n \mid^2, \tag{3.3}$$

where $||$ denotes the usual complex absolute value.In particular,$N_1 = Z(1)$.

We have,

$$(q^{n/2} - 1)^{2g} \leq N_n \leq (q^{n/2} + 1)^{2g},$$

i.e., N_n is asymptotically of magnitude q^{ng}.

This result agrees with the Hasse theorem stated in [17].

Hasse Theorem: *Let $\#E(F_q)$ be the order of an elliptic curve (g=1) group $E(F_q)$ defined over a field F_q with q elements. Then*

$$(\sqrt{q}-1)^2 \leq \#E(F_q) \leq (\sqrt{q}+1)^2$$

Here the explicit formulas in the simplest cases g=1 and g=2.

g = 1: If α is a root of $T^2 + M_1 T + q$,then $N_n = \mid 1 - \alpha^n \mid^2$.

Example: The equation E: $y^2 = x^3 + x + 6$ over the finite field Z_{11} (the integers modulo 11) defines an elliptic curve.

Here $q = 11$, $n = 1$ and $\#E(F_{q^n}) = 12$. Therefore $M_1 = 12 - 11 = 1$. This gives $N_1 = 13$ which includes the point at infinity.

g = 2: We first find the number of $\mathbf{F}_q$ and $\mathbf{F}_{q^2}$ solutions of $v^2 + h(u)v = f(u)$, thereby determining M_1 and M_2. The coefficients of $Z(T)$ are given by $a_1 = M_1$, $a_2 = (M_1^2 +$

Next, let γ_1 and γ_2 be the two roots of the quadratic equation $X^2 + a_1 X + (a_2 - 2q)$ $= 0$. Then α_j for $j = 1, 2$ is a root of the quadratic equation $X^2 - \gamma_j X + q = 0$.

Finally, $N_n = \mid 1 - \alpha_1^n \mid^2 \mid 1 - \alpha_2^n \mid^2$.

For cryptographic purposes it is desirable for $N_n = \#(\mathbf{J}(\mathbf{F}_{q^n}))$ to be divisible by a large prime number(about 30–40 decimal digits). The ideal case is for N_n itself to be prime.However, this rarely happens, because $\mathbf{J}(\mathbf{F}_{q^d})$ is a subgroup of $\mathbf{J}(\mathbf{F}_{q^n})$ for any divisor d of n, and so $N_d \mid N_n$.

Definition: We say that N_n is almost prime if N_n divided by the least common multiple of N_d $(1 \leq d < n, d \mid n)$ is prime. In particular, for n prime we say that N_n is almost prime if N_n/N_1 is prime.

We now assume that n is prime. By (3.3), we have

$$N_n/N_1 = \prod_{j=1}^{g} \mid (1 - \alpha_j^n)/(1 - \alpha_j) \mid^2 . \qquad (3.4)$$

If the α_j are not all conjugates, i.e., if the polynomial $Z(T)$ factors over the rationals, then even for n prime the value N_n/N_1 in (3.4) has a corresponding factorization.

Therefore, we have to look for an irreducible Z(T) for obtaining a large prime factor in the order of the jacobian.

3.5 Cryptosystems

As the HDLP is intractable, we can construct various public key cryptosystems in which taking large multiples of a group element is the trapdoor function.

In the following subsections, we shall discuss the hyperelliptic curve analogs of some public key cryptosystems discussed earlier.

3.5.1 Diffie-Hellman Scheme

Suppose that two users A and B want to agree upon a secret key.Then the Diffie-Hellman key exchange [7] works as follows:

Private keys:

- m_A (the private key of user A)
- m_B (the private key of user B)

Public Information:

- the finite field $\mathbf{F}_{q^n}$
- the equation of curve C
- a fixed element $D_0 \in \mathbf{J}(\mathbf{F}_{q^n})$
- $m_A D_0$ (the public key of user A)
- $m_B D_0$ (the public key of user B)

Each user A chooses a large integer m_A, which is kept secret, and computes and makes public the divisor $m_A D_0$. When two users A and B wish to have a key for use in some other cryptosystem, they use the divisor $m_A m_B D_0 \in \mathbf{J}(\mathbf{F}_{q^n})$. Here divisors are reduced to the form $\mathrm{div}(a,b)$ with $\deg b < \deg a \leq g$. Some standard way may be agreed upon, using the coefficients of a and b, to associate to $\mathrm{div}(a,b)$ an integer which serves as the key.

3.5.2 ElGamal Scheme

Consider a hyperelliptic curve C defined over a field $\mathbf{F}_{q^n}$ with order of its jacobian as $\#\mathbf{J}(\mathbf{F}_{q^n})$. Let $D_0 \in \mathbf{J}(\mathbf{F}_{q^n})$ be a fixed and publicly known divisor. Each user chooses an integer m_i randomly, such that, $0 < m_i < \#\mathbf{J}(\mathbf{F}_{q^n})$ and makes the divisor $m_i D_0$ public while keeping m_i secret. The working field and the equation of the curve are made public.

Now we have to first embedd the message onto some point of the curve before going to the actual scheme.

Message Imbedding

Here is one possible probabilistic method to imbed plaintexts as points on a curve C defined over $\mathbf{F}_{q^n}$ .Let k be a large enough integer so that we are satisfied with a failure probability of 1 out of 2^k when we attempt to imbed plaintext message units m; in practice k=30 or at worst k=50 should suffice. We suppose that our message units m are integers $0 \leq m \leq M$. We also suppose that our finite field is chosen so that $q^n > Mk$. We write the integers from 1 to Mk in the form $mk + j$,where $1 \leq j \leq k$, and we set up 1-1 correspondence between such integers and a set of elements of $\mathbf{F}_{q^n}$. For example, we write such an integer as an r- digit integer to the base q, and take the r digits, considered as elements of Z/qZ, as the coefficients of a polynomial of degree $r-1$ corresponding to an element of $\mathbf{F}_{q^n}$, i.e., the integer $(a_{r-1}, a_{r-2}, ..., a_0)_q$ corresponds to the polynomial $\sum_{i=0}^{r-1} a_j X^j$,which, considered modulo some degree$-r$ irreducible polynomial over $\mathbf{F}_q$,gives an element of $\mathbf{F}_{q^n}$.

Thus, given m, for each $j = 1, 2, ..,k$ we obtain an element x of $\mathbf{F}_{q^n}$ corresponding to $mk + j$.

Let C has equation $v^2 + h(u)v = f(u)$, as before. Choose the coordinate u=x as described in the previous paragraph and attempt to solve $v^2 + h(x)v = f(x)$ for v.

Case 1. q is odd. Then the problem reduces to taking a square root in a finite field. There is approximately a 50% chance that a solution v=y exists. If no solution exists, then we choose another u=x as described earlier and repeat the procedure.

Case 2. q is even.Then $h(x) \neq 0$, and the change of variables $z=v/h(x)$ leads to the equation $z^2 + z = a$, where $a = f(x)/h(x)^2$. This equation has a solution $z \in \mathbf{F}_{q^n}$ if $Tr_{\mathbf{F}_{q^n}/\mathbf{F}_2} a = 0$ and does not have a solution if this trace is 1. In the latter case, we must choose another $u = x$ and start again. In the former case we can find z as follows:

If $q = 2^n$ is an odd power of 2, simply set $z = \sum_{j=0}^{(n-1)/2} a^{2^{2j}}$.

For even n, first choose γ such that $Tr_{\mathbf{F}_{q^n}/\mathbf{F}_2}\gamma$=1. Next, set $\partial_j = a + a^2 + a^4 + ... + a$ for j = 1,2,...,n. Finally, take $z = \sum_{j=1}^{n} \partial_j \gamma^{2^{j-1}}$.

Now that we have mapped the message to point (x,y) on the curve C, we can map it to the divisor D_m as usual.

ElGamal Procedure

Suppose user A has to send a message D_m to user B,$D_m \in \mathbf{J}(\mathbf{F}_{q^n})$. A chooses a random integer m_A such that $0 < m_A < \#\mathbf{J}(\mathbf{F}_{q^n})$ and sends the following pair of points as the ciphertext to B.

$$C_m = (m_A D_0, D_m + m_A(m_B D_0))$$

Here $m_B D_0$ is the public key of B.

To decrypt the message, B multiplies $m_A D_0$ with his secret key m_B and finds the opposite of this divisor. He then adds this to the second divisor in the ciphertext pair to get the original message.

$$D_m = D_m + m_A(m_B D_0) + (-m_B(m_A D_0))$$

It is clear that there is a message expansion by a factor 2. For each message divisor, two divisors have to be sent as ciphertext. From the security point of view, same m_A should not be used for consecutive blocks of encryption. If m_A is used for more than one block, knowledge of one block D_{m_1} of the message enables an intruder to compute other blocks as follows. Let C_1 and C_2 be two consecutive cipher blocks for the messages D_{m_1} and D_{m_2}. Using the same m_A,

$$C_1 = (m_A D_0, D_{m_1} + m_A(m_B D_0))$$

$$C_2 = (m_A D_0, D_{m_2} + m_A(m_B D_0))$$

Now, if D_{m_1} is known, the common key $m_A m_B D_0$ can be easily found, therby revealing the plaintext corresponding to C_2.

A single curve C defined over a field $\mathbf{F}_{q^n}$ alogwith the base divisor D_0 can be shared by a group of people.The security of the system depends on the hyperelliptic curve discrete logarithm problem.

3.6 Implementations over F_{2^n}

Curves over $\mathbf{F}_{2^n}$ are advantageous for the following reasons:

- The arithmetic in $\mathbf{F}_{2^n}$ is easier to implement in computer hardware than the arithmetic in finite fields of characteristic greater than 2.
- When using a normal basis representation for the elements of $\mathbf{F}_{2^n}$,squaring a field element becomes a simple cyclic shift of the vector representation,and thus reduces the multiplication count in adding two points.

3.6.1 Normal basis representation

The field $\mathbf{F}_{2^m}$ can be viewed as a vector space of dimension m over $\mathbf{F}_2$. That is, there exists a set of m elements $\alpha_0, \alpha_1, ..., \alpha_{m-1}$ in $\mathbf{F}_{2^m}$ such that each $\alpha \in \mathbf{F}_{2^m}$ can be written uniquely in the form

$$\alpha = \sum_{i=0}^{m-1} a_i \alpha_i, \textit{ where } a_i \in \{0, 1\}.$$

We can then represent α as the $0-1$ vector $(a_0, a_1, ..., a_{m-1})$. In hardware, a field element is stored in a shift register of length m. Addition of field elements is performed by bitwise XOR−ing the vector representations,and takes one clock cycle.

In general, there are many different bases of $\mathbf{F}_{2^m}$ over $\mathbf{F}_2$. A *normal basis* of $\mathbf{F}_{2^m}$ over $\mathbf{F}_2$ is a basis of the form

$$\{\beta, \beta^2, \beta^{2^2}, ..., \beta^{2^{m-1}}\},$$

where $\beta \in \mathbf{F}_{2^m}$; it is well known [16] that such a basis always exists.Given any element $\alpha \in \mathbf{F}_{2^m}$, we can write $\alpha = \sum_{i=0}^{m-1} a_i \beta^{2^i}$, where $a_i \in \{0, 1\}$. Since squaring is a linear operator in $\mathbf{F}_{2^m}$,we have

$$\alpha^2 = \sum_{i=0}^{m-1} a_i \beta^{2^{i+1}} = \sum_{i=0}^{m-1} a_{i-1} \beta^{2^i} = (a_{m-1}, a_0, ..., a_{m-2}),$$

with indices reduced modulo m. Hence a normal basis (NB) representation of $\mathbf{F}_{2^m}$ is advantageous because squaring a field element can then be accomplished by a simple rotation of the vector representation, an operation that is easily implemented in hardware; squaring an element also takes one clock cycle.

Multiplication in a normal basis element representation is more complicated. Let the elements A, B, C of $\mathbf{F}_{2^m}$ in terms of NB be given as :

$$\begin{aligned} A &= \sum_{i=1}^{m-1} a_i \beta^{2^i}, \\ B &= \sum_{i=1}^{m-1} c_i \beta^{2^i}, \\ C &= A.B \\ &= \sum_{i=1}^{m-1} c_i \beta^{2^i} \\ &= \sum_{i=1}^{m-1} \sum_{j=1}^{m-1} a_i b_j \beta^{2^i} \beta^{2^j} \end{aligned}$$

Let the cross product terms:

$$\beta^{2^i} \beta^{2^j} = \sum_{k=1}^{m-1} \lambda_{ij}^{(k)} \beta^{2^k}, \lambda_{ij}^{(k)} \in \{0, 1\}$$

Substitution yields the following bilinear form for c_k:

$$\begin{aligned} c_k &= \sum_{i=1}^{m-1} \sum_{j=1}^{m-1} \lambda_{ij}^{(k)} a_i b_j \\ &= \sum_{i=1}^{m-1} \sum_{j=1}^{m-1} \lambda_{ij}^{(0)} a_{i+k} b_{j+k} \end{aligned}$$

where the subscripts of a and b are taken mod m. Thus we have

$$c_0 = \overline{A} \wedge \overline{B}^T; \wedge = (\lambda_{ij}^{(0)})$$

$$\begin{aligned} \overline{A} &= (a_0, a_1, ..., a_{m-1}) \\ \overline{B} &= (b_0, b_1, ..., b_{m-1}) \\ \overline{C} &= (c_0, c_1, ..., c_{m-1}) \end{aligned}$$

where $\overline{B}^T$ is the transpose of $\overline{B}$. The remaining coefficients of C can be found using the same matrix, but with $\overline{A}$ and $\overline{B}$ cyclically shifted.

In terms of hardware implementation of the arithmetic operation of multiplication the circuit to compute c_0 also computes c_k if the registers holding $\overline{A}$ and $\overline{B}$ are cyclically shifted k positions to the left. Massey and Omura constructed a serial−in serial−out multiplier to exploit this particular aspect of normal bases.

The *complexity* of such a circuit is determined by C_N ,the number of non−zero terms $\lambda_{ij}^{(0)}$,since this quantity measures the number of interconnections between the registers containing A,B and C. Clearly, we have $C_N \leq m^2$. A lower bound on C_N is $C_N \geq 2m - 1$ [28].If $C_N = 2m - 1$,then the normal basis is said to be *optimal.* In [28], constructions are given ,together with a list of fields for which these bases exist.

We see that the addition of divisors requires squaring of a polynomial in the given field. If the normal basis representation is used for the coefficients of the polynomial then the addition of divisors can be carried in an efficient manner.

In this chapter we have made our acquaintance with the hyperelliptic cryptosystems. The requirements for a secure cryptographic system over hyperelliptic curves are discussed. The effectiveness of theory is shown through implementations described in the next chapter.

Chapter 4

Results and Conclusion

4.1 Introduction

The calculation of modular products are an important part of software implementations of many cryptographic protocols such as the well–known RSA public key algorithm. In these exponential cryptographic systems, there is a need for fast modular exponentiation, that is the calculation of

$$C = M^e \ mod \ n \tag{4.1}$$

where for acceptable levels of security C, M, e and n are multiprecision numbers. Similarly, the basic operation performed in a hyperelliptic cryptosystem is the computation of multiplicity, mD, of a divisor on the curve defined over a finite field. Multiplication and squaring in (4.1) is replaced by addition and doubling in our case. For a large n and m, the time complexity of elementary operations as well as the number of elementary operations are very high. Thus, reducing the number of such operations is important when implementing the above algorithms.

Let us first consider the computation $C = M^e \ mod \ n$. The conventionally used algorithm for this is the binary method, described by Knuth [10]. The method is outlined below.

G is finite group of order n.

Input: $\alpha \in G$, $l \in Z$

Output: α^l

1. Let $l = \prod_{i=0}^{t} b_i 2^i$, $b_i \in \{0, 1\}$, $b_t = 1$, be the binary representation of l.
2. Set $\beta \leftarrow \alpha$.
3. For i from $t - 1$ down to 0 do

$$\beta \leftarrow \beta.\beta$$

   If $b_i = 1$ then $\beta \leftarrow \beta.\alpha$

4. Output β

We see that the algorithm comprises of a sequence of squarings and multiplications, the total number of which depends on $v(e)$, the number of 1's(hamming weight) in the binary representation of the exponent. In this method for each bit of the exponent except the first, squaring is done and if the bit is 1, then a multiplication is also done.

In our case, algorithm for computing mD can be given as:

Input: $D \in \mathbf{J}$,$m \in Z$.

Output: mD

1. Let $m = \prod_{i=0}^{t} b_i 2^i$, $b_i \in \{0, 1\}$, b_t=1, be the binary representation of m.
2. Set $D' \leftarrow D$
3. For i from $t - 1$ down to 0 do

$$D' \leftarrow D' + D'$$

   If $b_i = 1$ then $D' \leftarrow D + D'$

4. Output D'

As we know from our algorithm for addition of divisors, that doubling a divisor is cheaper(in terms of operations) as compared to the addition of two distinct divisors, we can successfully apply the above method for the computation of mD.

In practice, rather than using the pure repeated doubling method, in some cases it is more efficient to combine it with a procedure which replaces mD by a linear combination with small integer coefficients of the divisors D^{σ^j} (where $\sigma : x \to x^q$ is the Frobenius automorphism of $\mathbf{F}_{q^n}$). This sometimes reduces the total number of additions of divisors that must be performed in order to compute mD. Koblitz in [12] described a procedure which in many cases simplifies the computation of large multiples of group elements. We now give the theorem stated by him.
Let $\mathbf{J}$ be the jacobian of genus g curve defined over $\mathbf{F}_q$. Suppose that n_0 is large enough so that

$$(1 + q^{-n_0/2})^{2g} < 2.$$

Then there exists a polynomial–time algorithm which for any m and n gives an expression for mD, $D \in \mathbf{J}(\mathbf{F}_{q^n})$; *in the form* $\sum_{j=0}^{n-1} a_j D^{\sigma^j}$ *in which the integers* a_j *satisfy* $\mid a_j \mid < q^{n_0 g}$.
Our aim is to resolve mD into $\sum_{j=0}^{n-1} a_j D^{\sigma^j}$ accordig to the constraints laid by Koblitz's theorem.
For this proceed in the following manner :

Input : The polynomial Z(T) associated with the curve C, along with its complex conjugate roots, i.e. ,

$$Z(T) = \prod_{j=1}^{g} (T - \alpha_j)(T - \overline{\alpha_j}).$$

Output : An integer n_0 .
Steps :
Initially $n_0 = 1$. If the constant term $q^{n_0 g}$ of Z(T) is greater than the sum of the absolute values of all other coefficients then Stop else find n_0 as given in (1).

1. Find n_0 such that

$$(1 + q^{-n_0/2})^{2g} < 2$$

2. Find $Z_{n_0}(T)$ given as

$$Z_{n_0}(T) = \prod_{j=1}^{g}(T - \alpha_j^{n_0})(T - \overline{\alpha_j}^{n_0})$$

3. Express $D \in \mathbf{J}$ as

$$q^{n_0 g} D = -(Z_{n_0}(\sigma^{n_0}) - q^{n_0 g})D,$$

where $\mathrm{Z}(\sigma) \in \mathrm{Z}[G]$ and $G = \mathrm{Gal}(\mathbf{F}_{q^n}/\mathbf{F}_q) = \{\sigma^j\}_{j \in Z/nZ}$.

The results obtained on implementation of this algorithm have been discussed in the next section.

4.2 Implementation

The effectiveness of the various algorithms mentioned earlier has been shown by implementing them in SIMATH.

SIMATH is a software package which facilitates computations using algebraic structures and number theory. It was developed in Germany.

Besides other feature, it consists of a polynomial package for computation with polynomials in any number of unknowns over any of the structures contained in the arithmetic package.

4.2.1 Construction of irreducible Z(T)

In Chapter 3, Section 3.4, an algorithm for finding an irreducible polynomial Z(T) of degree $2g$ associated with the curve C was presented. Especially g=1, 2 cases were discussed.

We have selected certain curves conforming to the equation of hyperelliptic curve (3.1) and found the polynomial Z(T) associated with them. Tables 4.1 and 4.2 list the curve C along with the associated polynomial Z(T) for genus two and one

	Equation of C	$\#C(F_2)$	$\#C(F_{2^2})$	M_1	M_2	$Z(T)$
1.	$v^2+v=u^5+u^3$	4	4	2	0	$T^4+2T^3+2T^2+4T+$
2.	$v^2+v=u^5+u^3+1$	0	4	-2	0	$T^4-2T^3+2T^2-4T+$
3.	$v^2+v=u^5+u^3+u$	2	8	0	4	T^4+2T^2+4
4.	$v^2+uv=u^5+1$	3	3	1	-1	T^4+T^3+2T+4
5.	$v^2+uv=u^5+u^2+1$	1	3	-1	-1	T^4-T^3-2T+4

Table 4.1: Genus two curves over $\mathbf{F}_2$ with irreducible Z(T).

	Equation of C	$\#C(F_2)$	M_1	$Z(T)$
1.	$v^2+v=u^3$	2	0	T^2+2
2.	$v^2+v=u^3+u$	4	2	T^2+2T+2
3.	$v^2+v=u^3+u+1$	0	-2	T^2-2T+2
4.	$v^2+uv=u^3+u^2+1$	1	-1	T^2-T+2
5.	$v^2+uv=u^3+1$	3	1	T^2+T+2

Table 4.2: Genus one curves over $\mathbf{F}_2$ with irreducible Z(T).

respectively. Here we listed only those curves for which Z(T) was irreducible, i.e., it did not factorize over the rationals.

4.2.2 Utilizing Koblitz's algorithm for computing mD

We tried to resolve mD into $\sum_{j=0}^{n-1} a_j D^{\sigma^j}$ according to the algorithm given in previous section.

	Equation of C	n_0	$Z_{n_0}(T^{n_0})$	Expression
1.	$v^2+v=u^5+u^3$	6	$(T^{12}+64)^2$	$64D=-D^{(12)}$
2.	$v^2+v=u^5+u^3+1$	6	$(T^{12}+64)^2$	$64D=-D^{(12)}$
3.	$v^2+v=u^5+u^3+u$	1	T^4+2T^2+4	$8D=D^{(6)}$
4.	$v^2+uv=u^5+1$	5	$(T^{10}-2T^5+32)$	$2^{10}D=-D^{(20)}+9D^{(15)}$
			$\cdot(T^{10}-7T^5+32)$	$-78D^{(10)}+288D^{(5)}$
5.	$v^2+uv=u^5+u^2+1$	5	$(T^{10}+7T^5+32)$	$2^{(10)}D=-D^{(20)}-9D^{(15)}$
			$\cdot(T^{10}+2T^5+32)$	$-78D^{(10)}-288D^{(5)}$
6.	$v^2+v=u^5$	1	T^4+4	$4D=-D^{(4)}$

Table 4.3: Expressions for computing mD in case of genus two.

	Equation of C	n_0	$Z_{n_0}(T^{n_0})$	Expression
1.	$v^2+v=u^3$	1	T^2+2	$2D=-D^{(2)}$
2.	$v^2+v=u^3+u$	4	T^8-16	$16D=D^{(8)}$
3.	$v^2+v=u^3+u+1$	4	T^8+8T^4+16	$4D=-D^{(4)}$
4.	$v^2+uv=u^3+1$	4	T^8-T^4+16	$16D=D^{(4)}-D^{(8)}$

Table 4.4: Expressions for computing mD in case of genus one.

Tables 4.3 and 4.4 list the results obtained for genus two and one respectively. Now we reduce m (in case of mD) to a list of modulo 64 for the firstcurve of Table 4.3.

$$m=\sum_j m_j 64^j,\ 0\le m_j<64,$$

and

$$mD=\sum_j(-1)^j m_j D^{\sigma^{12j}}.$$

We now present the results which compare the time taken by pure doubling method and the method proposed by Koblitz for the computation of mD.

m	Pure doubling method	Proposed method
2^6	10 ms.	8 ms.
2^{18}	40 ms.	14 ms.
2^{54}	120 ms.	31 ms.
2^{66}	150 ms.	36 ms.
2^{72}	170 ms.	39 ms.

Table 4.5: Average time for computation of mD on the stated curve

We did the computations over $\mathbf{F}_{2^{37}}$. The irreducible modulo polynomial in special bit notation, the polynomials *h(u), f(u)* of curve C and the chosen divisor D_0 (comprising of polynomials *a(u)* and *b(u)*) in sparse representation are given below in the same order :

(Consider a polynomial $a_n x^n + a_{n-1} x^{n-1} + ... + a_0$ defined over F_{q^n}. (1) Its special bit representation comprises of a list in which the first element denotes the degree of polynomial which is n in our case, and the remaining is the value of the polynomial obtained by replacing x by q. (2) Its sparse representation comprises of a list of the kind $(n(a_n)\ n-1(a_{n-1})\ ...\ 0(a_0))$ where the coefficients $(a_n\ a_{n-1}...a_0)$ are represented in special bit notation.)

(37 128 748589809)

(0 (0 1))

(5 (0 1) 3 (0 1))

((1 (0 1) 0 (1 2)) (0 (34 28 959279377)))

The timing details have been given in Table 4.5. If we employ the normal basis representation for the coefficients of the polynomials, we will be having $64D$ in 12 cyclic shifts.

n	order
5	13·61
17	13·1326700741
23	13·5415624023749
29	13·22170214192500421
37	13·1453030298001690873541
41	13·2953·125965976976392564317
43	13·59516319662966858346861 49
47	13·5641·270097268484167653999069
53	13·207973·300074592543931816180128 97

Table 4.6: Almost prime values of $\#\mathbf{J}(\mathbf{F}_{2^n})$ for C given by $v^2 + v = u^5 + u^3$

4.2.3 Computing the order of the jacobian

For a secure cryptographic system, the order of the jacobian should be divisible by a large prime factor.We now tabulate some results based on the algorithm for computing the order of the jacobian given in Section 3.4.

In the tables, the notation P(X) denotes a prime factor of X digits.

Note, in Tables 4.6–4.12, N_1, i.e., $\#\mathbf{J}(\mathbf{F}_2) = \mathrm{Z}(1)$.

For a secure cryptographic system, a prime factor of around 30 digits is needed in the order of the jacobian. In Tables 4.6–4.12 we see that genus one and two curves satisfy this requirement. The genus two curves give the same number of digits in the prime factor of the order of the jacobian as genus one curves but with small extensions of GF(2) as compared to the latter. Therefore for genus two curves, computations have to be done for comparatively small fields.

n	order
5	2·2·2·101
7	2·2·2·1471
29	2·2·2·59·610759905970679
31	2·2·2·373·1545462843139867
47	2·2·2·247588030689074564787501495 1
67	2·2·2·2·13·17·19·64831124924907666407179433147680231 7
73	2·2·2·3·31·17293·693323770117758392170552824804496113 7

Table 4.7: Almost prime values of $\#\mathbf{J}(\mathbf{F}_{2^n})$ for C given by $v^2 + uv = u^5 + 1$

n	order
5	2·701
7	2·11173
13	2·33651853
19	2·137987565589
29	2·59·2442221041332503
43	2·38685599708692613136777013
47	2·1787·55419806385021374142200 63
59	2·2·2·71549·58055842540542424418569440081 1

Table 4.8: Almost prime values of $\#\mathbf{J}(\mathbf{F}_{2^n})$ for C given by $v^2 + uv = u^5 + u^2 + 1$

n	order
5	1321
7	14449
11	4327489
19	275415303169
23	70334392823809
31	4611545283086450689
53	10177·797186200486710330329346259 3

Table 4.9: $\#\mathrm{J}(\mathbf{F}_{2^n})$ for C given by $v^2 + v = u^5 + u^3 + 1$

n	order
29	5·107367629
43	5·1759217765581
53	5·1801439824104653
89	5·123794003928545064364330189
107	5·857·378668090616600572642192533 97
173	5·13625405957·P(42)
233	5·3108221·P(63)
239	5·77852679293·P(61)

Table 4.10: $\#\mathrm{J}(\mathbf{F}_{2^n})$ for C given by $v^2 + v = u^3 + u$

n	order
47	140737471578113
73	9444732965601851473921
89	1069·579017791994999956106149
137	189061·92152570791184058739061733088636270 1
139	557·125116389129996763586027250922976428790 9
239	P(72)

Table 4.11: $\#\mathbf{J}(\mathbf{F}_{2^n})$ for C given by $v^2 + v = u^3 + u + 1$

n	order
73	2·877·5384682420498870703
101	2·12676506002282308861428085080 11
107	2·81129638414606692182851032212511
109	2·32451855365842670148744865646 1467

Table 4.12: $\#\mathbf{J}(\mathbf{F}_{2^n})$ for C given by $v^2 + uv = u^3 + u^2 + 1$

4.2.4 Implementation of ElGamal scheme

The ElGamal scheme has been implemented in software. The software uses three files namely, *publ_key.elg*, *priv_key.elg*, *hyp_cur.elg*. The file *publ_key.elg* contains the public keys and *priv_key.elg* has the secret key of the user. The file *hyp_cur.elg* contains the parameters of the hyperelliptic curve over which the system works. At the start of the program, the curve parameters are read from this file. By changing the parameters in the curve, the system can be made to work with different set of curves. This makes the system very flexible. Shown below is a sample of the file *hyp_cur.elg*. Here we have taken a hyperelliptic curve defined over $\mathbf{F}_{2^{37}}$. The modulo polynomial, the curve equation given by *h(u)* and *f(u)*, and the divisor D_0 comprising of *a(u)* and *b(u)* are given below in that order :

(37 128 748589809)
(0 (0 1))
(5 (0 1) 3 (0 1))
((1 (0 1) 0 (1 2)) (0 (34 28 959279377)))

The software consists of the following modules :

- Encryption
- Decryption
- Key Generation

We took an arbitrary point on the curve and mapped it to a divisor. This constituted our message divisor. We then successively passed it through the encryption and decryption modules. Our aim is to recover back the original message divisor.

Key Generation

Any new user has to get registered before he can enter the system. The user chooses an USER−ID and enters a random number. This random number serves as the users secret key k. The program now computes the public key $k \cdot D_0$ for this user. The public key and the secret key are written in the files *publ_key.elg* and *priv_key.elg*

respectively. Given below is a sample session for key generation.

The parameters of the working curve are :

n=37

genus=2

the modulo polynomial is

(37 128 748589809)

h(u)= (0 (0 1))

f(u)= (5 (0 1) 3 (0 1))

Base divisor is ((1 (0 1) 0 (1 2)) (0 (34 28 959279377)))

press any key to continue

Enter :

'1' for encryption

'2' for decryption

'3' for key generation

'4' for listing valid ID's

< 3 >

Please enter the USER−ID you require in alphanumeric < *Neha* >

Enter a random number < $\#J(F_p)$; This is your secret key. .

< 9494384751792551490964 >

The public key 'k.D' of Neha is

((2(0 1) 1(34 20 592015758) 0(30 1 246137796)) (1(35 37 773824283) 0(36 67 457521479)))

Key generation successful!!!

press any key to continue

A sample of the file *priv_key.elg* is shown below :

#Neha

94943847517925514909 64

#Ruchi

98786512345078123456789

#Akash

34567887654312344321122

#Mayank

45454556566767787889890

We have applied Koblitz's method for computing $k.D$. Therefore the private key of each user is first broken into a list of ascending powers of 64.The lists obtained are :

#Neha : (20 22 31 20 24 37 5 37 14 4 43 0 2)

#Ruchi : (21 36 21 2 45 20 16 44 1 41 51 58 20)

#Akash : (34 49 7 23 50 16 17 20 15 55 30 20 7)

#Mayank: (34 19 13 16 5 39 14 56 48 34 1 40 9)

A sample of the file *publ_key.elg* is shown below :

#Neha

((2(0 1) 1(34 20 592015758) 0(30 1 246137796)) (1(35 37 773824283) 0(36 67 457521479)))

#Ruchi

((2(0 1) 1(36 76 229191970) 0(35 39 659078824)) (1(6 111 224047393) 0(36 123 54102510)))

#Akash

((2(0 1) 1(34 16 453924485) 0(35 53 299556516)) (1(4 24 855898293) 0(34 23 128317278)))

#Mayank

((2(0 1) 1(35 53 743168203) 0(36 101 21483127)) (1(5 32 163998377) 0(36 127 546219480)))

Now consider a sample session between Neha and Ruchi. Neha wants to communicate to Ruchi. So she selects the latter's public key and multiplies it with her

own private key. To the common key generated, she now adds the message divisor and sends it to Ruchi. The latter sees the public key of Neha, multiplies it with her own private key and then computes the opposite of the generated common key. She now adds it to the received divisor to recover the message divisor.

press any key to continue

Enter :

'1' for encryption

'2' for decryption

'3' for key generation

'4' for listing valid ID's

< 1 >

Enter your identity : < *Neha* >

ID check O.K.

Enter receiver's identity : < *Ruchi* >

ID check O.K.

ENCRYPTING!!!

Encryption process successful.

The group element sent to Ruchi is

((2(0 1) 1(36 127 309624909) 0(36 78 722981934)) (1(35 48 729765714) 0(36 113 747156001)))

press any key to continue

Enter :

'1' for encryption

'2' for decryption

'3' for key generation

'4' for listing valid ID's

< 2 >

Enter your identity : < *Ruchi* >

ID check O.K.

DECRYPTING!!!

Decryption process successful.

press any key to continue

<Type any key other than '1', '2', '3', and '4' to exit>

4.3 Conclusion

In this thesis we looked into various aspects of implementing a hyperelliptic cryptosystem. We saw that the protocols implemented over hyperelliptic curves have the advantage of having small keys than the conventional systems defined over multiplicative group of a finite field for the same level of security.

We generated curves with large prime factor in the order of the jacobian and discussed technique for efficient computation of multiplicity of a divisor. The efficiency of Koblitz's method over the pure repeated doubling method in computing mD will be more if we employ normal basis representation for the coefficients of the polynomials.

We paid special attention to fields with characteristic two. Similar results could be carried over to curves over prime fields, though the speed will be slow in this case. In our work, all computations have been shown for genus two case. As the genus increases, the computational complexity increases too. However, genus one and two curves can be efficiently used for the construction of secure cryptosystems.

For the implementation of a full fledged hyperelliptic cryptosystem, all the basic blocks have to be highly optimized, e.g., basic field arithmetic routines are to be written in assembly language. Our implementation was optimized only at the algorithm level.

Bibliography

1. S.S.Abhyankar. *Algebraic Geometry for Scientists and Engineers.* AMSpublications, 1984.

2. E.R.Berlekamp. *Algebraic coding theory.* New York: McGraw Hill, 1968.

3. D.Cantor, "Computing in the jacobian of an hyperelliptic curve", Math. Comp, 48, 95-101, 1987.

4. J.S.Chahal. *Topics in number theory.* Plenum Press, 1986.

5. H.Cohn. *A classical invitation to algebraic numbers and classfields.* Springer-Verlag Universitext, 1978.

6. D.E.Denning. *Cryptography and data security.* Addison-Wesley, 1982.

7. W.Diffie and M.E.Hellman, "New directions in cryptography", IEEE Trans. on Information Theory, Vol.IT-22, No.6, Nov.1976, 644-654.

8. T.Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", IEEE Trans. on Information Theory, 31, 1985, 469-472.

9. W.Fulton. *Algebraic curves.* Benjamin, New York, 1969.

10. D.Knuth. *The art of computer programming.* Vol.2, Reading, MA: Addison-Wesley, 1981.

11. N.Koblitz, "Elliptic curve cryptosystems", Math. Comp., 48, 1987, 203-209.

12. N.Koblitz, "Hyperelliptic Cryptosystems", Journal of Cryptology, Vol.1, 1989, 139-150.

13. N.Koblitz. *Introduction to elliptic curves and modular forms.* Springer-Verlag, GTM No-97, 1984.

14. M.D.Mandelbaum, "On iterative arrays for the euclidean algorithm over finite fields", IEEE Trans. on Computers, Vol-38, No.10, Oct1989.

15. R.J.McEliece. *Finite fields for computer scientists and engineers.* Kluwer Academic Publications, 1992.

16. A.Menezes. *Applications of finite fields.* Kluwer Academic Publications, 1994.

17. A.Menezes. *Elliptic curve public key cryptosystems.* Kluwer Academic Publications, 1994.

18. A.M.Odlyzko, "Discrete logarithms and their cryptographic significance", Advances in cryptology: Proceedings of Eurucrypt'84, Springer-Verlag, New York, 1985, 224-314.

19. T.Okamoto, K.Sakurai, "Efficient algorithms for the construction of hyperelliptic cryptosystems", Advances in Cryptology-Crypto'91, LNCS, Vol.576, Springer Verlag, New York, 1991.

20. T.Ono. *An introduction to algebraic number theory.* 1988.

21. F.Oort. *Reducible and multiple algebraic curves.* Royal VanGorcum Ltd., Netherlands.

22. R.Rivest, A.Shamir, L.Adleman, "A method for obtaining digital signatures and public key cryptosystems", Comm. ACM.21, 2(Feb1978), 120-126; reprinted in Comm. ACM.26, 1(Jan1983), 96-99.

23. H.E.Rose. *A course in number theory.* Oxford University Press, 1994.

24. H.Shizuya, T.Itoh, K.Sakurai, "On the complexity of hyperelliptic discrete logarithm problem", Advances in Cryptology-Eurocrypt'91, Springer Verlag, New York, 1991, 337-351.

25. J.H.Silverman. *The arithmetic of elliptic curves.* Springer Verlag, GTM 106, New York, 1986.

26. M.A.Tsfasman, S.G.Vladut. *Algebraic-Geometric codes.* Kluwer Academic Publications, 1991.

27. S.A Vanstone, A.J.Menezes, T.Okamoto, "Reducing Elliptic curve logarithm to logarithms in a finite field", Proc. of STOC'91, 1991, 80-89.

28. R.Wilson, R.Mullin, I.Onyszchuk, and S.Vanstone, "Optimal Normal bases in $GF(p^n)$ ", Discrete Applied Mathematics, 22(1988/89), 149-161.

.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....